SEL COTS Study

Phase 1 Initial Characterization

Study Report

August 1998

Contents

Appendix B

Appendix C

Acronyms

References

Figures

Tables

Executive Summary

A current trend in software development, in both government and industry, is a move toward commercial off-the-shelf (COTS)-based software development. The Flight Dynamics Division (FDD)* at the National Aeronautics and Space Administration's (NASA's) Goddard Space Flight Center (GSFC) is also experiencing this shift in the way that satellite ground support software is being developed. The FDD has had a well-defined, documented process in place for many years that covers the area of standard software development. The logical step at this stage in the FDD's evolution is to document its COTS-based, or package-based, development processes.

The task of this Software Engineering Laboratory (SEL) COTS study was to evaluate to what degree the recommendations of *SEL Package-Based System Development Process* (which was defined based on the traditional SEL-recommended development process, modified by information on COTS development from the literature) have been implemented within the FDD and to determine what difficulties and successes FDD developers have had integrating COTS. While planning the COTS study, the study team concluded that the information needed was not readily available in the SEL database; therefore, alternative methods for data collection needed to be considered. The need for additional data forms to be filled out by project personnel was apparent, as well as a need to collect qualitative data through structured interviews. Interviews were conducted prior to considering new data forms so that insights gained from the interviews would be used in the design of the data forms. The interviews conducted are described in detail in Section 3.2.

More than 50 COTS packages were used by one or more of the 15 projects contacted during this study. Several COTS packages, such as Satellite ToolKit, were used by various projects. Some projects are using one or two COTS products, while others used more; one project used as many as 13 individual COTS packages in its system.

For the developers of COTS-based systems to integrate the components, several behind-the-scenes groups are crucial. Management support is needed with regard to decisions about which COTS to evaluate and use and which products to procure. The developers also rely on support from the procurement team. In many cases, a team would designate a point of contact with the vendor. Having the team involved with the vendor throughout the life cycle is valuable. Support from outside groups (e.g., projects with prior experience, the COTS evaluation team) enabled teams to more readily become proficient with a new product. However, the time and effort involved in learning about new products need to be planned for, whether it is the project coming up to speed or that an outside group, such as the COTS Evaluation Team, is investigating the product.

One contribution of this study is a detailed process characterization, which will serve as a baseline from which process improvements can be measured and described. Although not every team interviewed followed all of the steps outlined below, a composite process flow emerged from the interview data. Note: None of the project teams interviewed had begun sustaining engineering.

* The FDD ceased to exist as an organizational entity when GSFC reorganized in December 1997. Much of the work formerly performed by FDD (including SEL work) continues in various GSFC organizations.

This step will be evaluated in future studies. The steps in the overall process, also shown in Figure 5–1 and described in Section 5.1, are as follows:

- Requirements analysis
- Package identification, evaluation, and selection
- Non-COTS development
- Glueware requirements and development
- System integration and test
- Target system installation and acceptance test
- Discrepancy resolution
- Sustaining engineering

Another major contribution of the COTS study is a new set of data collection forms that will allow the SEL to collect data that correctly describes COTS-based development activities. These forms were developed in response to a need for more COTS-related data and to update the types of data maintained in the SEL database. This new set of forms was created by modifying the existing Weekly Effort Form and adding a new COTS and Tools Information Form. These forms appear in Appendix B and are described in Section 5.2.

This document includes recommendations for improvements to the FDD COTS-based software development process, based on the study of this process. Some of these are "local" improvements, i.e. small changes to the current way COTS-based development is carried out to make this process more efficient. Others are more "global" in the sense that they represent more sweeping changes to the process that should be further studied. These recommendations, described briefly here, are described fully in Section 5.3:

- Local
  - Consider a new type of review, called a COTS review, held early in the process to review high-level decisions about which components will be COTS, which will be developed from scratch, and what glueware will need to be developed.
  - Recommend more administrative support for procurement.
  - Rely on and support resource teams, such as the COTS Evaluation Team.
  - Produce a Quick Reference Guide for the COTS-based development process.
- Global
  - Avoid downstream conflicts between requirements and the capabilities of COTS packages by performing COTS identification, evaluation, and selection before requirements analysis.
  - Allow the software development team to make more of the decisions regarding COTS packages.
  - Adjust the COTS-based development process based on the amount of COTS used in the project.
  - Integrate the traditional development process used to build components and glueware with COTS-based development.

The SEL COTS study also introduced some methods new to the SEL for carrying out studies and disseminating findings to help the SEL respond to the changing, accelerating environment at NASA/GSFC. These methods are described in Section 5.4. For example, the use of qualitative data and analysis, in combination with other quantitative methods, has been found to be very useful. Also, *SEL Study Briefs* were introduced as part of this study to concisely document and quickly distribute timely information. These new approaches will be useful on future SEL studies, especially in light of the current pressures on the SEL to keep up with the rapidly changing nature of the development projects and organizations it is studying.

The future work of the COTS study is concentrated in three areas:

- Sustaining engineering of COTS-based systems
- Analysis of data from COTS-based projects to build cost models
- Risk analysis for COTS-based projects

These areas have been identified as having the most impact on the success of COTS projects. However, studying these issues would not be possible without the underlying understanding gained by the initial phase of the COTS study.

Section 1. Background

A current trend in software development, in both government and industry, is a move toward package-based software development. “Packages” refer to whole, completed software components or subsystems that have been built by an outside party. The intent is that these packages can be plugged into the system being developed, thus avoiding a lot of new development. In this way, the software development world is moving toward configuring systems from packaged components, rather than building whole systems from scratch. Several terms are used to refer to these packages.

- Commercial off-the-shelf (COTS) – packages purchased from a commercial vendor
- Government off-the-shelf (GOTS) – software developed for the Federal Government that is used by a team of developers other than the original authors of that software
- Package-based development – the most generic term, but not widely used among the technical community

COTS is the more commonly used term, and for the purpose of this study, it is being extended beyond its original meaning to include GOTS. Throughout this report, the terms “COTS” and “COTS-based development” are used, except when referring to *SEL Package-Based System Development Process* (Reference 1).

The Flight Dynamics Division (FDD)* at the National Aeronautics and Space Administration’s (NASA’s) Goddard Space Flight Center (GSFC) is also experiencing this shift in the way it develops satellite ground support software. The FDD has had a well-defined, documented process in place for many years that covers standard software development that has evolved and been optimized based on experience. The next logical step in the FDD’s evolution is to document its COTS-based development processes, analyze these processes, and improve them.

The NASA dictate to build all systems “faster, better, cheaper” in no way implies that methodology and process should be eliminated. Nevertheless, as applications are developed more rapidly, the process must change to suit the environment. In 1995 and 1996, FDD personnel were still learning how to develop software systems using COTS packages. Therefore, the Software Engineering Laboratory (SEL) relied on outside experiences to define a tentative process described in Reference 1. This document relied on a solid understanding of the FDD project domain, history, and environment to synthesize information from the literature into a strawman process to be used to produce COTS-based systems in the FDD. This initial strawman process was then reviewed for feasibility by key FDD software engineers (both civil servant and contractor) who had some experience with COTS. The resulting process was intended to be used on FDD projects that integrate COTS packages into their systems. As projects gained experience with the process, the SEL was refine it to reflect local experience and it was to be documented in an official supplement to *SEL Recommended Approach to Software Development*, Revision 3 (Reference 2).

* The FDD ceased to exist as an organizational entity when GSFC reorganized in December 1997. Much of the work formerly performed by FDD (including SEL work) continues in various GSFC organizations.

Reference 1 is based on input from the following sources:

- D. Boland, *A Proposed Guide for Package Integration by FDD Project Teams*
- D. Boland and D. Messent, *SEAS Package-Based Development Guidebook*
- Integrated Monitoring, Analysis, and Control COTS System (IMACCS) Team, *IMACCS System Implementation Process and Lessons Learned*
- Loral Federal Systems, *COTS Integration and Support Model*
- CSC Catalyst^{SM}, *Package-Based Development Methodology*
- B. Boehm, *COCOMO 2.0 Users Guide*
- Mitre Corporation, *The Impact of COTS on Maintenance Organizations*

The process defined was meant to be used in conjunction with Reference 2. On projects where the system was composed of both COTS products and custom-built components, Reference 2 was to be followed for the custom-built part and Reference 1 for the COTS part. It was fairly easy to align the two processes using the reviews as a guide. Table 1–1 lists the phases, major activities, products, and management checkpoints (such as reviews) that this process recommends. Because the package-based process was new, projects were asked to record lessons learned and provide feedback to the SEL so that the process could be improved for future projects.

The task of the SEL COTS study was to determine to what degree the Reference 1 recommendations have been implemented within the FDD and to learn about the difficulties and successes FDD developers have had integrating COTS. One original objective of this work was to produce a revision of Reference 1. However, this first study of COTS-based development in the FDD does not represent enough experience with this new development paradigm to warrant a revision of the document. After several planned follow-on studies, this document will be revised and made available in the same way as other SEL "best practices" documents, such as the *SEL Manager's Handbook.* Eventually, this new document will be merged with the next revision of Reference 2.

This report describes the major activities, findings, and contributions of the SEL COTS study, as well as plans for several follow-on studies intended to explore some specific COTS-related questions in more detail.

Table 1–1. Former SEL Package-Based Development Phase Characteristics

Phase	Major Activities	Products	Management Check-points
Requirements Analysis and Package Identification	■ Requirements analysis ■ COTS package survey and preliminary evaluation	■ Requirements ■ Strawman high-level architecture ■ Candidate packages	■ System requirements review (SRR)
Architecture Definition and Package Selection	■ Package evaluation ■ Requirements modification to use existing packages ■ Prototyping	■ Modified requirements ■ System architecture ■ Final packages	■ System design review (SDR)
System Integration and Test	■ Use-case implementation ■ Independent testing	■ Delivered system	■ User demonstrations ■ Operational readiness review (ORR)
Technology Update and System Maintenance	■ Evaluation of new products and technology	■ Enhanced system	■ User demonstrations

Section 2. Study Goals

The general goal of this SEL COTS study was to learn about how COTS-based development is carried out in the FDD and to provide recommendations for how it could be conducted better. The three original experimental objectives were as follows:

1. Revise the data collection for COTS-based projects.
2. Produce baseline models (e.g., effort, defects, schedule) for COTS-based projects in FDD, where possible, after collecting data from COTS-based project team personnel.
3. Revise *SEL Package-Based System Development Process* as needed after evaluating the experiences of COTS-based project team personnel who have evaluated, selected, and integrated COTS products into their software systems. The revised document should define a robust process that effectively works for COTS usage in Flight Dynamics so that the SEL can distribute this as a SEL-recommended process for COTS-based projects.

The first goal was achieved and is described in Section 5.2. The baseline models of the second goal need to wait until more data is collected using the new data collection mechanisms designed for COTS projects. However, a detailed baseline process model has been built and is described in Section 5.1. As discussed previously, the revision of the SEL document on COTS-based development (goal 3) has been deferred until more experience with COTS-based development in the FDD has resulted in more consistency and consensus on how it is carried out.

Section 3. Methods

This section describes the three basic research methods used to carry out the COTS study. Section 3.1 presents the goal/question/metric (GQM) structure that was used to plan the study, Section 3.2 describes the GQM structure that helped guide the interviews conducted, and Section 3.3 describes some of the quantitative data analysis.

3.1 Development of Goals, Questions, and Metrics

The various activities performed by the COTS study team are reflected in the team's six measurement goals:

1. Analyze the COTS products being used in the FDD for the purpose of identification, with respect to name and type, from the point of view of COTS-based project team personnel.
2. Analyze the COTS-based development process used in the FDD for the purpose of characterization, with respect to steps followed in their process, from the point of view of COTS-based project team personnel.
3. Analyze the COTS-based development process used in the FDD for the purpose of characterization, with respect to cost, schedule, problems encountered, risks, and guidance or documentation used, from the point of view of COTS-based project team personnel.
4. Analyze the COTS-based development process used in the FDD for the purpose of characterization of the differences from traditional development processes and from SEL COTS-based process from the point of view of COTS-based project team personnel.
5. Analyze the COTS-based development process used in the FDD for the purpose of evaluation in comparison to traditional development processes and to SEL COTS-based process, from the point of view of COTS-based project team personnel.
6. Analyze Reference 1 for the purpose of improvement, with respect to usefulness, from the point of view of COTS-based project team personnel.

Many of the questions formulated to address these goals (as part of the GQM process) were incorporated into the interview guides described in Section 3.2. Incorporation of these questions into the guides constitutes the metrics part of the GQM. Other questions were better addressed through analysis of data. Attempting to do this revealed the inadequacies of the current SEL data collection in terms of collecting information on COTS-based development. This, in turn, led to deferment of some of the associated goals and to redesign of some data forms (described in Section 5.2).

For example, goal 1 led to a question on the interview guide that asked project personnel which COTS products and types of COTS products were being used on projects. Information generated by this question led to a table listing all such COTS products, thus satisfying goal 1. Goal 2 led to a number of questions on the interview guides, addressing the steps followed in COTS-based development and the variations that occur in different situations. The result of collecting that information is the baseline process characterization described in Section 5.1. Goal 3 also led to some interview guide questions, in particular concerning problems and risks associated with COTS and the documentation used. Some of the results of posing these questions are presented in Section 4. Other questions related to goal 3 had to do with cost and schedule and required some data analysis to be answered. This led to redesigning some SEL data forms (described in Section 5.2).

Goals 4 and 5 also generated interview guide questions having to do with differences, advantages, and disadvantages of COTS-based development in comparison to traditional development. The comparison with Reference 1 was made by the study team because project personnel were generally not familiar with the SEL process document. These results are summarized in Sections 4 and 5.1.

Finally, goal 6 resulted in some questions that could not be addressed by this initial phase of the COTS study. Improvement to Reference 1 will have to wait until later stages of the study.

3.2 Interviews Conducted

While planning the COTS study, the study team concluded that the information it needed was not readily available in the SEL database; therefore, the team would have to consider alternative methods for data collection. The need for additional data forms to be filled out by project personnel was apparent, as well as a need to collect qualitative data through structured interviews. Interviews were conducted prior to developing new data forms so that insights gained from the interviews could be used in the design of the forms.

Basically, three different types of interviews were conducted, each at a different level of detail. The first set of interviews was intended to obtain basic information on which projects were using COTS products, which products were being used, and how the products were being used. The second set of interviews was aimed more specifically at finding the process steps that were being followed in COTS-based development, in addition to the advantages and disadvantages of COTS use. After an initial process description was derived from the second set of interview data, the third set of interviews was conducted to validate the team's understanding of the COTS-based development process. Interview guides were designed and used for each level of interview (see Appendix A).

The interviews were conducted by teams composed of two study team members. One team member served as the interviewer and the second as the scribe. The interviewer's responsibilities included conducting the interview using the interview guide as an outline, posing open-ended questions to the interviewee, and following up as appropriate to gather as much information as was reasonable.

The scribe's two main responsibilities were to (1) keep notes on the information covered in the interview to enable the interviewer to concentrate on facilitating the interview process and (2) document the interview in a structured textual manner following the basic outline of the interview guide. The scribe usually spent some time after the interview to write detailed notes on all of the interviewee's responses, based on notes taken during the interview.

Two quality checks are built into this system. First, at the end of the interview, the scribe asks the interviewee to clarify parts of the interview as needed and asks any questions from the interview guide that may have been inadvertently omitted by the interviewer. Second, prior to finalizing the interview notes, the interviewer reviews them for concurrence. In this manner, the two-person team functions more effectively than a single interviewer.

The interview data was analyzed in several ways to extract a variety of different types of information. Different study team members reviewed the interview notes, each concentrating on gaining an understanding of a different aspect of COTS-based development. The notes were reviewed using varying levels of rigor. For example, at the most basic level, a list was generated from the interview notes of all the COTS products considered by projects. A new data collection form—the COTS and Tools Information Form (CTIF)—designed to collect this information directly from projects in the future is now readily available from the SEL database, along with characteristics of the COTS product and its use on that project.

At the other end of the spectrum, the interview notes were also analyzed using a method loosely based on the constant comparison method (References 3 and 4), a rigorous qualitative analysis method used to identify trends and consensus in textual data. This type of analysis led to findings about the process steps followed in COTS-based development and the main advantages and disadvantages of using COTS. This analysis contributed to the design of a revised Weekly Effort Form (WEF). The new data forms are described in Section 5.2.

The study team learned several lessons about interviewing as a result of this process.

- By structuring the interviews and limiting them to 30 minutes, people were willing to make room in their schedules for the interviews. Of the 25 people contacted to request an interview, only one was unable to spare 30 minutes.
- The two-person interview team was effective. The checks and balances built into such a team are its greatest strength.
- When collecting qualitative data, verification steps are crucial. The interview process had two main verification steps:
  - Verification of the interview notes between the interviewer and the scribe
  - A third interview that verified the process seen and that uncovered any additional areas for concentration

  An additional verification step occurs when an *SEL Study Brief* (see Section 5.4) is posted on the Web—the contributors of information are contacted so that they review the brief and participate in the built-in *SEL Study Brief* feedback loop.

3.3 Metrics Data Analysis

The study team also analyzed historical quantitative data from projects. In particular, bar charts generated (see Figure 3–1) helped the team identify issues for further study. Quantitative data collected through the use of the new data forms will be analyzed as part of the future work of the COTS study team.

Section 4. Major Findings

More than 50 COTS packages were used by one or more of the 15 projects contacted during this study. Several COTS packages, such as the Satellite ToolKit, were used by various projects. Some projects are using one or two COTS products, while others used more. One project used as many as 13 individual COTS packages in their system.

COTS packages were used to build ground support systems, as well as assist in a platform transition from mainframe to workstation that all flight dynamics systems underwent. COTS used in this effort ran the gamut from a COTS product that completes the entire function of a telemetry processor, to fourth-generation languages that must be delivered with the system to maintain it.

During the interviews, most developers said that they did not follow Reference 1. However, when asked to detail the steps they did follow, those steps generally did fall within the basic structure of the SEL process. Although the personnel interviewed used different words to describe their processes, the study team could provide an integrated abstraction of a common COTS-based development process. Few of the developers interviewed described their processes in the way presented in Section 5.1, but on seeing that abstraction, they all could identify their activities within the abstracted process. This indicates that the process defined in Reference 1 is reasonable in this environment and that improvements to the document will be a matter of presenting the information in the most readily accessible and useful form and not a complete overhaul of the process.

It became apparent in the early stages of the COTS study that the data the SEL had been collecting would not be sufficient to gain insights into the COTS issue. Several lessons were learned as a result. The team learned that conducting interviews allowed collection of the data needed, conducting interviews provided the types of data that should be collected, and creating a new form and modifying an existing form would enable the team to translate data collected during an interview onto the forms. This allows future interviews to concentrate on the qualitative data.

The developers interviewed were asked to describe the major differences between COTS-based development and traditional development and the advantages and drawbacks. Some mentioned the obvious difference, i.e., that there is now a lot of software that does not need to be implemented. It is no longer the task of building a big system, but of using already-built pieces. Other less-obvious differences also were mentioned. Some of the opinions raised, although not necessarily supported by quantitative data, are as follows:

- Different design phases – Design focused more on how to fit pieces together rather than the internal workings of different modules
- Looser process requirements – Because much of the standard SEL recommended process did not apply to COTS-based development and schedules were very tight, project personnel felt freer to loosen the process requirements
- New or greatly increased need for vendor interaction – Interaction with the vendor occurred at different levels throughout the project
- Procurement skills now needed – Procurement required some technical knowledge so it was not an adminis-

trative activity, but technical personnel were not prepared to deal with procurement issues

- New or greatly increased need for product evaluations – Another new skill or activity that developers were not always prepared for
- No unit test or inspections of packaged software – Because source code is not delivered with most packaged software, it is not possible to unit test or inspect it
- Teams should be empowered to make decisions regarding COTS, as opposed to management dictates that a specific COTS shall be used

Advantages of COTS-based development that were mentioned included

- More flexible requirements – There was usually some room to adjust requirements to fit the COTS products being used
- Less process overhead – As above, tightened schedules and an undefined process allowed projects to cut out process steps that they felt were unnecessary, while still maintaining a sense of process
- Less code to write – Large portions of the system were constituted by COTS and thus did not have to be written
- Less debugging – Similarly, portions of the system constituted by COTS did not have to be debugged
- Shorter cycle time – Possibly because of schedule pressure, COTS projects seemed to be completed more quickly, although this needs to be confirmed empirically
- Better adherence to schedule – There was a perception that schedules were kept better in COTS projects, although this must be confirmed empirically
- Serendipitously useful functionality in COTS packages – Sometimes functionality was discovered in a COTS package that was useful, even though the project had not originally planned to use it

Disadvantages mentioned involved

- Dealing with the vendor – The vendor constituted one more party with whom communication channels had to be established and maintained
- Less than full knowledge beforehand of the product – Sometimes surprises occurred having to do with the quality or functionality of a COTS package
- Dependence on the vendor – Project personnel had to rely on the vendor for a variety of technical issues, but vendor personnel were not always as helpful or available as promised
- Vendor negotiations – Technical personnel were not always prepared to deal with the business aspects of purchasing COTS packages
- Looseness of the process – Also mentioned as an advantage, but some people thought that more rigor was needed

For the developers of COTS-based systems to integrate the components, several behind-the-scenes groups are crucial. Support from management is crucial. The developers also rely on support from the procurement team. In many cases, a team would designate a point of contact with the vendor. Having the team involved with the vendor throughout the life cycle is invaluable. Support from outside groups (e.g., projects with prior experience, COTS evaluation team) allowed teams to become proficient more readily with a new product.

Section 5. Major Contributions

The major contributions of the COTS study are several tools that will greatly enhance the further study of COTS-based development in the FDD and other environments at GSFC. These contributions also will facilitate the improvement of COTS-based development.

First is a detailed process characterization, which will serve as a baseline from which process improvements can be measured and described. Second is a new set of data collection forms that will allow the SEL to collect data that correctly describes COTS-based development activities. Finally, two sets of improvement recommendations—one for COTS-based development itself and another for the activities of the SEL—help the SEL respond to GSFC's changing, accelerating environment.

5.1 COTS-Based Development Process Baseline Characterization

As a first step in understanding where COTS-based development in the FDD stood, the study team analyzed the current data collection. Historically, the SEL collects effort data. For typical pre-COTS era projects, the SEL has a baseline of effort divided into four simple categories of activities. The SEL had earlier anticipated the need for data specific to COTS projects and had made an attempt to gather data on this effort, but the level of detail was too general to allow understanding of the COTS-related effort. One indication that the SEL was not capturing useful data is the large amount of effort that fell into the "other" category.

Clearly, the quantitative information available was not sufficient to identify and understand the new issues that were arising in relation to the use of COTS packages in FDD projects. To gather more and richer information on this topic, the study team designed and conducted structured interviews, using three levels of interview guides at increasing levels of detail, with representatives from 12 projects. Topics covered included the process steps carried out, what problems were encountered with the use of COTS in development, and how the incorporation of COTS has changed the software development process.

Reference 1 splits the package identification and package selection steps, the first being part of requirements analysis and the second being part of architecture definition, with the SRR in between. However, the interviews show that the process being followed within the FDD generally combines package identification, evaluation, and selection and performs all three at one time, after requirements analysis. This is probably because, in many projects, the requirements and sometimes even the system architecture are defined outside the scope of the development project. Therefore, often the only activity necessary before completing the high-level design is deciding which parts of the system will be COTS-based (and on which COTS they will be based) and which will be written in-house. Another difference between the COTS process characterized and the 1996 SEL process is that the 1996 process requires more reviews (e.g., SRR, SDR, ORR) than are actually being carried out. Some projects conduct SRRs and SDRs, but many do not. Most of the projects studied have not yet reached the ORR stage.

When asked what types of documentation or information would be helpful to projects attempting a package-based development, several things were mentioned. Projects need guidance on how to integrate the traditional style development of glueware into the process, but do not need as much guidance on product evaluation because they can rely on other parts of the organization for that (e.g., previous project personnel or the COTS Evaluation Team). Several people mentioned that a streamlined, condensed process document (as compared the current *SEL Package-Based System Development Process*) would be helpful. Also helpful would be up-to-date lists of local resources in terms of people with experience with different packages and vendors, available site licenses, and where to look for help with the overwhelming administrative tasks involved in package-based development.

The interviews uncovered the new process flow, shown in Figure 5–1. The study team discovered more complexity in the current practice than expected in theory. For example, the team had expected vendor interaction to be simple and to end with the purchase of a product. In reality, the interaction continues throughout the life cycle, and the flow of information is not merely one way. Surprisingly, the team found a strong dependence on bi-directional information flow. Also shown is a more constant involvement with separate organizations, such as other projects that also use COTS, independent evaluation teams, and other customers of the vendor. Portions of the COTS-based systems include traditionally developed software. Therefore, an issue to consider is how to fit together the SEL's traditional process, as documented in Reference 2, and its new way of doing business by integrating COTS packages to build a system.

The software development teams interviewed included both FDD and Computer Sciences Corporation (CSC) personnel. Although not every team followed all eight steps, a composite process flow emerged from the interview data. The contribution of modeling this process flow is that it constitutes an abstraction of the process at a level of detail that is high enough to represent all the different ways that COTS-based systems are being developed in the FDD, but with enough detail to provide useful guidance to COTS projects. Note: None of the project teams interviewed had begun step 8. This step will be evaluated in future studies. The eight steps in the overall process, as shown in Figure 5–1, are as follows:

1. Requirements analysis
2. Package identification, evaluation, and selection
3. Non-COTS development
4. Glueware requirements and development
5. System integration and test
6. Target system installation and acceptance test
7. Discrepancy resolution
8. Sustaining engineering

The earliest steps in COTS-based development are similar to traditional development—requirements gathering. In the requirements phase, a strong emphasis is on gathering external information. Much of this information comes from separate organizations, particularly the product vendor, in the form of documented functionality. Some project requirements are predefined, with minimal requirements analysis needed. Early reviews of the requirements are crucial even with a less formal process.

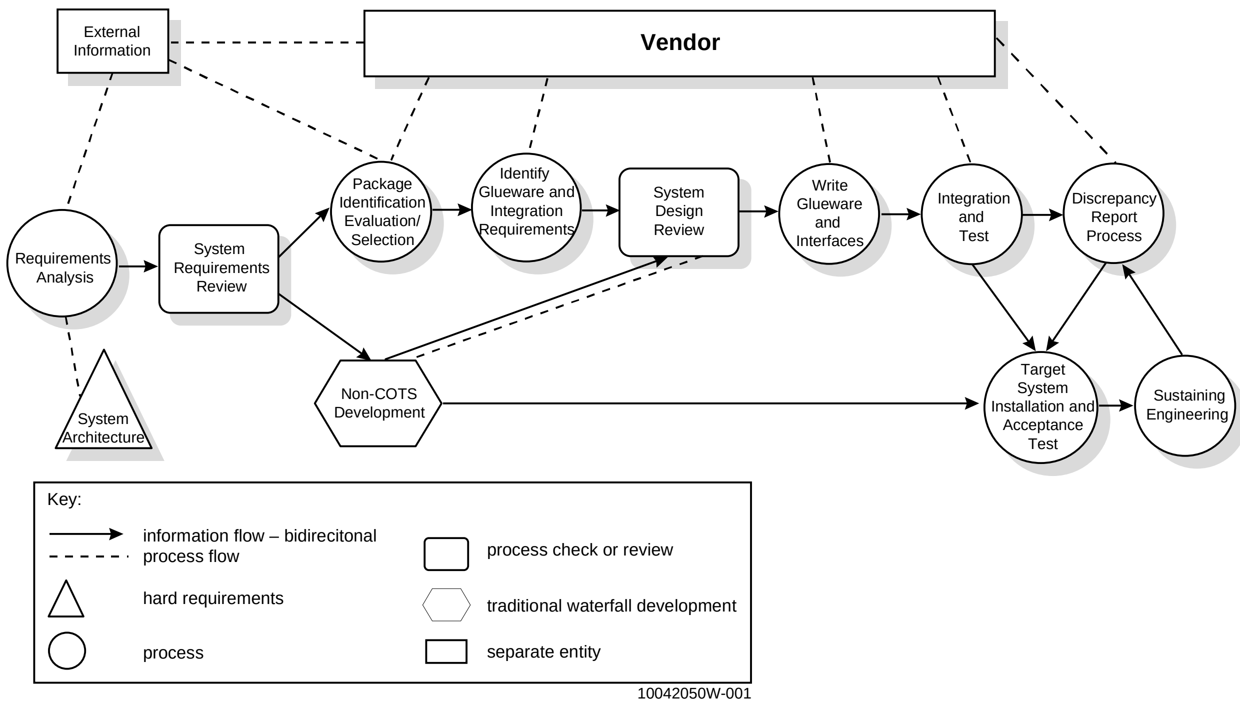

Figure 5–1. New SEL Process Flow

Following requirements analysis are the new and concurrent steps of package identification, evaluation, and selection. These new activities require new technical skills and new administrative duties, especially in the area of procurement.

Package identification consists of Web searches, product literature surveys and reviews, other system component reuse, and recommendations from external sources. Product information is kept in either a central justification notebook or an evaluation notebook. Not only are product evaluation notes kept, but also subjective comments concerning the vendor quality and responsiveness.

As packages are identified, the evaluation and selection processes begin. Package evaluation steps mentioned in the interviews consisted of prototyping, vendor demonstrations, and in-depth review of literature, such as manuals and users guides. Glueware and interfaces as dictated by the system architecture, operating system, and hardware are identified. Vendor training, sites, and availability are considered. Procurement issues surface, such as development fees for added requirements, licensing and maintenance fees, and sustaining engineering support.

Step 2 sometimes uses a weighted average. Vendor capabilities are listed and mapped to the system requirements. With team agreement, weights of importance are assigned to each requirement. Each team member then votes. Team members are polled and the votes tallied. Discussion ensues and a choice is made. In cases where the vendor will code additional functionality, the vendor is notified of the decision. In one case, when the team told the vendor it was selected, the vendor announced a hidden cost. Negotiations ended altogether, and the second choice vendor and package were used.

In both of these first two process stages, the study team found that some projects relied on the COTS Evaluation Team, which is chartered by the parent organization to survey the marketplace and evaluate vendor packages that fall within the domain expertise of the mission team's organization. The evaluation team then reports its findings and offers this knowledge to the project teams. The project team is ultimately responsible for deciding which package to select and integrate. The evaluation team can be important when delivery time is driving the project—time the development team does not have for product evaluations.

Most projects studied have an element of traditional development that does not depend on COTS or other packages. This development begins in parallel with the early COTS-related steps, as a traditional development project. Non-COTS cost and schedule are monitored. Bi-directional information flow between the COTS-based process flow and the non-COTS development comes into play in the design review. Only some teams held a formal SDR, but all teams had some kind of mechanism to apprise the customer of the design.

After the design review, whether it is formal or informal, traditional non-COTS development continues in parallel with the coding of the glueware and the interfaces. Close contact with the vendor technical staff or a competent help desk is essential during this development.

The integration step varies a great deal from project to project, depending on which and how many COTS products are being used. At system integration and testing, the COTS packages are treated as black boxes. The teams commented that testing focused on the interface glueware and the input file format. Again, the importance of the vendor technical staff or help desk availability was emphasized. Testing is conducted on each software component as the components are integrated piece-by-piece.

Unlike the traditional life cycle, no formal acceptance testing or operational readiness reviews were mentioned by the teams. The development team installs the software on the target system. Once installed, navigational training to familiarize the customer with the system is conducted. During this phase, a member of the development team is the single point of contact or intermediary between the customer and the vendor. This person is responsible for reporting discrepancies and handling software “patches” or corrections. Interviewees mentioned that software patches were placed on vendor Web sites that were downloaded to the target system.

The end of the configuration process is marked by step 8, sustaining engineering. To date, no team interviewed had reached this step.

5.2 New Data Collection Forms

In response to a need for more COTS-related data, the SEL realized an opportunity to update the types of data maintained in the SEL database. This was accomplished by modifying the existing WEF and adding a new CTIF. These forms also served to codify and quantify some of the information the study team found to be important from the interviews conducted. By using the new forms, this information does not have to be included in future interviews, but can be collected in more cost effectively.

5.2.1 Weekly Effort Form

As the interview data lead the study team to define the COTS-based development process, the team discovered that projects were conducting new activities:

- *COTS/GOTS evaluation* activities included identifying packages, collecting information, attending demonstrations, and evaluating and selecting COTS/GOTS packages.
- *COTS/GOTS integration* included integrating COTS and GOTS, possibly with other software components, to produce individual applications or subsystems. This also included the writing and debugging of glueware.
- *COTS package familiarization* is spending time to learn to use a COTS package, not including formal training, which would be included under other effort categories, or package familiarization for the purposes of evaluation.
- *Configuration management* had not previously been a separate category.
- *Procurement* included procuring and purchasing packages, interacting with vendors regarding licensing and maintenance agreements, etc.

In June 1997, these new activities were merged into the WEF (the SEL form in use since October 1995) for collecting effort data from the technical personnel. This merger created a WEF modified for COTS that was then used on a trial basis by two projects. Appendix B provides copies of the October 1995 WEF and the June 1997 experimental COTS WEF. After experimental use of this COTS WEF and a few resulting updates, the SEL decided to implement the updated WEF across the organization. This was accomplished through full consultation with FDD technical personnel. The resulting WEF was put into place November 1997 (also included in Appendix B).

Figure 5–2 shows the type of data collected by the experimental COTS WEF, the October 1995 WEF, and the even earlier SEL WEF that was in use prior to October 1995. The leftmost bar shows the typical distribution of effort on completed FDD projects prior to October 1995. The major activities fall into four groups: *design*, *code*, *test*, and *administrative*; no activity deals with COTS.

The middle bar shows the effort distribution for a nearly complete FDD project that was developing during the era of the October 1995 WEF that involved some COTS integration. The October 1995 WEF introduced a *predesign* category. It also introduced a *technical other* category that contained a prototyping activity plus the single *COTS* activity (a catchall for any COTS-related work). For the purposes of this study, the term *technical other* is used to clarify a category listed in the October 1995 and June 1997 WEFs as *miscellaneous*. This clarification has been formalized in the November 1997 WEF. Note that this middle bar shows a great increase in the proportion of project effort spent in the *administrative* activity. Various hypotheses were examined to explain this change, but none proved conclusive.

The rightmost bar shows the distribution of effort for a FDD project that involved a fair amount of COTS integration, but was only partially complete. This project began using the experimental COTS WEF soon after the project began. Only about 12 weeks of data were available for analysis. The data in this bar is thus insufficient to draw any conclusions on the distribution of effort on a typical FDD project, yet alone a project in another environment. Data on several complete projects would be required before the typical FDD effort distribution on a COTS project could be determined. However, Figure 5–2 provides some idea of how the new WEF changes the way effort on COTS projects can be viewed.

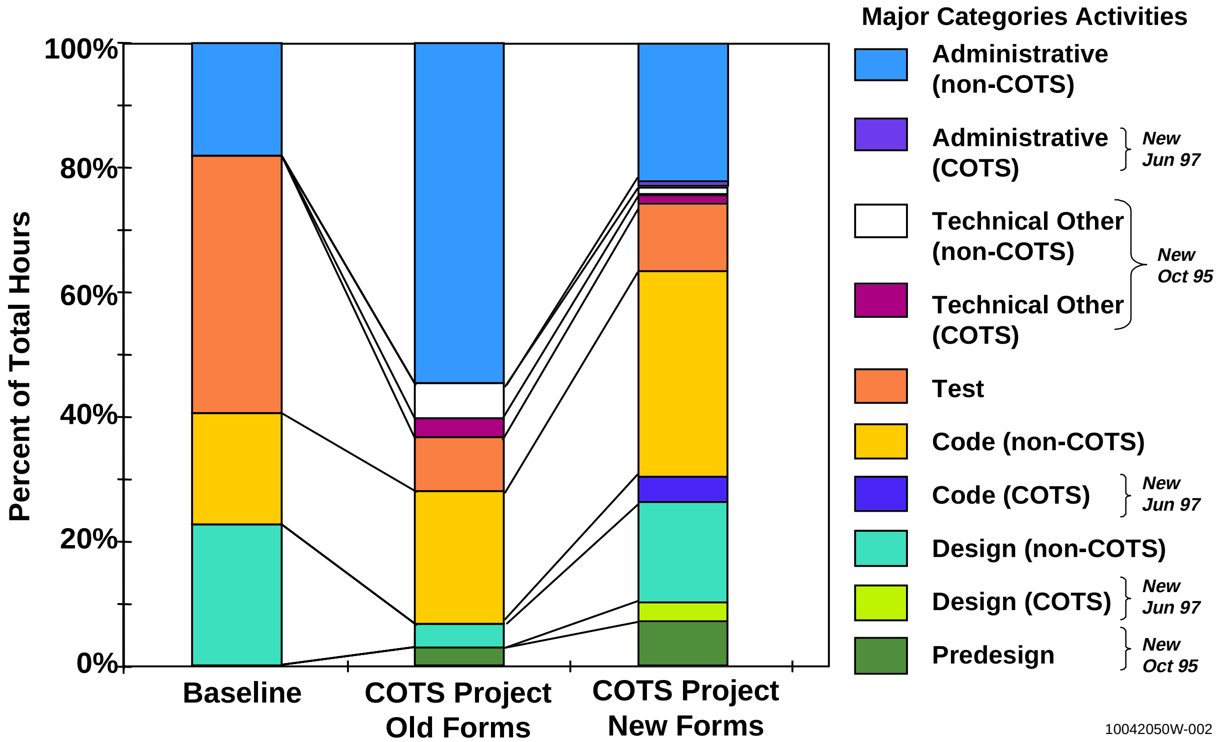

Figure 5–2. Data Collected by Experimental COTS WEF

5.2.2 COTS and Tools Information Form

To collect context data about the COTS packages used on projects, the SEL developed the CTIF (shown in Appendix B). The need for the CTIF became evident during the interview process. The study team was collecting qualitative data, such as which COTS packages are used, what support is provided by the vendor, and whether COTS is embedded into the system or is merely a tool. Rather than maintaining all this information in the interview notes, the team developed the CTIF to collect data that would be stored, and thus readily accessible, in the SEL database. Using the CTIF to collect this context data allows the study team to characterize the COTS products to better compare projects that are related either in the type of COTS products used or in functionality provided by COTS.

5.3 Recommendations for Process Improvement

This section outlines some recommendations for improvements to the COTS-based software development process in the FDD, based on this study of the process. These recommendations are divided into two groups:

- A set of *local* improvements, i.e., small changes to the current way COTS-based development is carried out to make this process more efficient.

- A set of *global* recommendations, which represents more sweeping changes to the process that should be further studied

5.3.1 Local Improvements

Based on this study of current COTS projects, several local, or relatively narrow scope, improvements to the process are recommended. The first has to do with one of the process steps, the design review. The second and third improvements are organizational rather than technical. The fourth improvement is a way to provide better support to personnel carrying out COTS projects.

- Improvement 1 – Most COTS projects perform some sort of review after a high-level design is complete and before implementation and integration begin. Many projects treat this as a type of design review, although other topics are covered, such as the decisions made about the COTS products to be used. The study team recommends that more guidance be given in conducting this review, which means designing a new type of review—a COTS review. This review would go over the high-level decisions about which components will be COTS, which will be developed from scratch, and what glueware needs to be developed.
- Improvement 2 – More administrative support for procurement is needed, especially easily accessible records about what products have already been procured by the organization and where they are located.
- Improvement 3 – Increase reliance on and support for the COTS Evaluation Team. Evaluation by individual projects tends to be narrow in scope, concentrating only on those packages with which project team members are familiar.
- Improvement 4 – A "Quick Reference Guide" for the COTS-based development process would be useful. Such a guide should be brief (no more than two sides of a sheet of paper) and highlight the most important aspects that a COTS project team needs to keep in mind. The COTS study team is currently developing such a guide, which soon will be available to FDD projects.

5.3.2 Global Recommendations

Several areas require more detailed study. These areas could potentially improve the effectiveness of the process and are suggested by the findings of this study, but it is not clear at this time how or whether they should be implemented. These suggestions include the following:

- Perform COTS identification, evaluation, and selection before or in conjunction with requirements analysis, and then write requirements tailored to the COTS products chosen. This would help eliminate later problems caused by conflicts between requirements and the capabilities of the COTS packages. It would also maximize the benefits of the COTS packages.
- Allow the software development team to be included in the decisions regarding COTS packages. For many projects, COTS choices were made outside the team, thus ignoring the team's expertise and experience. Requirements also often came from outside the team, and conflicts between requirements and COTS func-

tionality often occurred later in the project.

- Adjust the COTS-based development process based on the amount of COTS used in the project. A project that only uses one COTS package to implement a minor subsystem should not follow the exact same development process as a project that is integrating numerous COTS packages that will constitute most of the resulting system.
- Integrate the traditional development process used to build components and glueware with COTS-based development. Figure 5–1 shows basically two parallel tracks in the process flow, representing two separate types of development taking place in the same project. These parallel tracks may imply duplicated effort and communication difficulties between project members. Integrating the two tracks might increase effectiveness and efficiency.

These points are not independent. The first two both have to do with the relationship between requirements and COTS choices. The second two are both related to the balance between COTS integration and from-scratch development, either of components or glueware. Clearly, these are not straightforward to implement and thus require further study.

5.4 Recommendations for SEL Improvement

The SEL COTS study also introduced some methods new to the SEL for carrying out studies and disseminating findings. These new approaches will be useful on future SEL studies, especially in light of current pressures on the SEL to keep up with the rapidly changing nature of the development projects and organizations it is studying.

5.4.1 Interviews and Qualitative Data

Empirical studies in software engineering, like the ones that the SEL has engaged in for 2 decades, have traditionally relied on standard quantitative methods to characterize some aspect of a software development process. In some cases, several quantitative studies of various sizes and scopes have been conducted to address one general issue, e.g., Cleanroom software development [see *Evaluation of Software Technologies: Testing, Cleanroom, and Metrics* (Reference 5)]. Approaching a problem from several angles in this way yields a more complete description of a particular process or of the effect of a particular technology. This approach has helped the SEL and other organizations learn a great deal about their software business. In recent years, however, software projects in the SEL environment have become both more complex and faster-paced, as is true in much of the software industry. This has motivated the SEL to find ways to provide richer answers to more complex problems in less time.

One approach to achieving this goal is for the SEL to use different research methods than it is accustomed to using, in particular qualitative methods. Qualitative data is information in the form of words and pictures, as opposed to quantitative data, which is in the form of numbers. Qualitative analysis is simply the examination and analysis of qualitative data to form conclusions and hypotheses. Qualitative data is by definition richer and carries more information than quantitative data. On the other hand, it is more complex and harder to analyze. Qualitative analysis methods have been designed to deal with this complexity

(References 3 and 4). Combinations of qualitative and quantitative methods are especially useful because the two types of methods tend to deal with the complexity of the subject in complementary ways.

The COTS study is one of the first SEL studies to use qualitative data to a large extent (Reference 6 provides another example). The qualitative data used in this study comes from extensive interviews with software developers and managers. Using this data has allowed an in-depth examination of COTS-based development that incorporates a variety of perspectives in one study. For example, data was collected on the problems encountered during COTS-based development, the different steps involved, the parts of the process that are effort-intensive, and the roles that must be filled to carry out this type of development. Much of this information would be very difficult to collect quantitatively and would have required multiple studies, each measuring various attributes in different ways.

The drawbacks to doing qualitative study is that it does not provide hard results in terms of easy-to-use mathematical models (e.g., regression models) or easy-to-summarize relationships between variables (e.g., correlations). Instead, qualitative results are messier (i.e., more complex) to reflect the complexity of the problem being described. It is usually helpful to combine qualitative and quantitative methods in the same study because the two approaches have complementary approaches to handling the complexity of the subject.

Quantitative approaches tend to abstract away complexity to reveal any strong relationships that might otherwise be obscured. Qualitative approaches, on the other hand, delve directly into the complexity that complicates the straightforward quantitative findings and attempt to bring some order to the complexity without simplifying it. The COTS study, at this point, suffers a bit from a lack of quantitative analysis of COTS-related data because very little data is available at this time. The study team's findings will be greatly strengthened and more useful when it has analyzed the data collected using the new data forms for several complete projects.

Working with qualitative data is also very effort intensive. Qualitative analysis cannot be automated in the same way as quantitative analysis through the use of statistical software. However, the kind of intensive study of the COTS process that the study team conducted to gain an initial understanding of a new phenomenon does not have to be done on a continual basis. Further study of the process can be conducted quantitatively, using the new data forms, possibly occasionally augmented with interviews to track the evolution of the process.

Qualitative data, mostly from interviews, is also being used to some extent on other ongoing SEL studies. In combination with other quantitative methods, the team believes the use of qualitative analysis in current and future studies will help the SEL provide the development community with more useful, in-depth, and realistic explanations of software development phenomena.

5.4.2 Study Briefs

The SEL realized a need for compact products because larger process documents were not being used by technical personnel. Time is of the essence, and if they need information, they talk to an expert and search the Web for others' experiences. The SEL, as the experience factory, sees as its responsibility to serve as a base of knowledge that can be tapped into. *SEL Study Briefs* are an example of this as they concisely document and quickly distribute timely information. A study brief is less than a process document, yet much more than informal communications. The modularity of the study briefs allows the user community to incorporate one page's worth of process into their busy schedules. Study briefs also serve as a tool for communication with the inclusion of the technical community in the feedback loop section. The study team took advantage of electronic media by putting the *SEL Study Briefs* on an internal Web page and using E-mail as the feedback mechanism. Additionally, the information gained throughout a study can be documented in the form of a study brief, and at the conclusion of the study, applicable study briefs can be incorporated into a study report such as this one. Table 5–1 shows the content of a *SEL Study Brief*. A sample study brief is shown in Appendix C.

Table 5–1. SEL Study Brief

Title	Explanation
Study Brief Number:	number assigned (in order of posting on Web)
Issue:	topic of study brief
Purpose:	goal of study brief
Current Understanding:	body of the study brief, may vary in style
Feedback:	comments received from audience after study brief has been posted to Web site
Original Author(s):	person or persons originating study brief
Responsible Author(s):	person responsible for receiving feedback, and possibly modifying the study brief (in most cases, same as original author)
Contributors:	others who contributed to the study brief
References/Relevant Links:	materials used in preparation of study brief, additional information on a topic, or hot links to online sources
History:	first published date, any revision dates

Section 6. Further Work

This initial phase of the COTS study was valuable not only in making the contributions outlined in this document, but also in clarifying the issues related to COTS-based development that are likely to have an impact on its success. A number of such issues could not be addressed in this phase of the study for various reasons, but will be addressed in the next phase.

A major issue that is crucial to understanding and ensuring the long-term success of COTS-based development is how COTS-based systems are maintained over time (i.e., sustaining engineering). The cost savings and other advantages of incorporating COTS into new systems may be overshadowed or even wiped out if such systems are difficult and costly to maintain. On the other hand, the difficulties of integrating COTS may pay off in maintenance if the high use of COTS actually makes maintenance easier. It is not clear which of these situations will prevail or in what situations COTS will be a maintenance asset or liability. This issue could not be studied in the initial phase of the COTS study because none of the COTS projects identified had completed development and entered into a sustaining engineering phase. However, many of these projects are now approaching the sustaining engineering, or maintenance, stage and would be ideal candidates for study in the next phase.

One of the original goals of the COTS study was to build baseline models for such development variables as cost, effort, and defects. However, the study team very quickly realized that the data currently collected in the SEL was not adequate for building such models, at least in the area of effort. This led to the redesign of some data collection forms, as described in Section 5.2. These new forms are the infrastructure that will facilitate the collection of data that can then be used to build quantitative models for cost and effort. Because few of the projects studied had started testing their systems, analysis of defect data will also have to wait until enough experience has amassed, in the form of collected data, on which to base defect models.

Another issue that was raised again and again during interviews with COTS-based project personnel is the various risks that are associated with this type of development. Although there is no clear characterization of these risks nor a set of risk mitigation strategies, there is a great deal of qualitative data from project personnel that identifies some COTS-related risks. This data needs to be analyzed, more data needs to be collected, and investigation needs to be conducted into strategies for avoiding and minimizing these risks.

Clearly, much is left to learn about COTS-based development. The study team has identified several issues that need further examination before it has an understanding of how and when COTS-based development succeeds in the FDD. However, studying these issues would not be possible without the underlying understanding gained by the initial phase of the COTS study described in this document.

Appendix A. COTS Interview Guides

Interview Guide 1a: Initial project interviews

Who: Project leads

Subjects covered: Background and current status of project, GSS versus MATLAB decisions, initial COTS information

Duration: 30 to 45 minutes

Note: This interview should also include introducing yourself and our study to the project leads.

Interviewee:

Interviewer:

Scribe:

Date of interview:

Duration:

Location:

Questions:

1. What is/are your ROLE(s) on this project?]Obtain both official titles, such as user, domain expert, and a description (e.g., technical or administrative, level of involvement).]
2. What is the current status of FDSS development for this project? What are the different applications being developed? Which have begun, are in progress, or are completed? [Gradually narrow down to attitude applications.]
3. For each application, how is it being developed? Using GSS and UIX? Using some COTS product such as MATLAB or STK? Did any modifications need to be made to the COTS or GOTS products? Describe the modifications and how they were made.
4. What deployment, development, or integration process did you use to produce these applications? Where did this process come from? What process documentation or guidance did you use, if any?
5. Are you aware of the *SEL Packaged-Based System Development Process* document?
6. Did you follow the *SEL Packaged-Based System Development Process* document?
7. Is there anything that we can do to make this a more useful, easier-to-follow process?
8. How were the decisions to use these COTS and GOTS products made? What were the steps in the decision process? What were the criteria?
9. Were lessons learned recorded? Where?
10. What types of problems did you run into with the COTS and GOTS products you chose?

11. What do you think are the biggest risks associated with these decisions? [Try to get a mapping between the criteria mentioned in question 3 and the risks mentioned here.] For example,
    - Unacceptable performance of the application
    - Reliability of COTS products
    - Delays waiting for something from another group
    - Delivered application is unmaintainable
    - Required skills not available
    - Key personnel leaving or being pulled off project at crucial points
    - Cultural clashes between personnel from different areas
    - Turnaround time for error fixes or added functionality
12. Any creative ways to protect against these risks?
13. What data did you collect during the project regarding COTS?
    - schedule
    - cost
    - errors
    - standard SEL data
14. What metrics do you see as valuable in managing COTS-based projects?
15. Was there a purchasing leader for this project? Who? (Discuss purchasing decisions and procurement.)
16. What other projects do you know are using or planning to use COTS, GOTS, or other package-based products?
17. Can I be put on your project mailing list and/or could I have access to your project Web page? What else would help me keep track of how the project is going? Where can I look at project documentation?
18. Who are the other core team members and what are their roles?

Interview Guide 2: COTS Follow Up interviews

Who: COTS-based project leads

Subjects covered: Follow-up COTS information

Duration: 30 to 45 minutes

Note: This interview should also include reintroducing yourself and our study to the project leads.

Interviewee:

Project(s):

Interviewer:

Scribe:

Date of interview:

Duration:

Location:

Date of initial interview:

Questions:

1. What did you do for the following? [Try to capture major activities, process, products, and reviews.]
   - Requirements analysis
   - Package identification
   - Architecture definition
   - Package selection
   - System integration
   - Test
   - Maintenance
2. What are the biggest differences between traditional development and package-based development?
3. What are the advantages of package-based development in comparison with traditional development?
4. What are the disadvantages of package-based development in comparison with traditional development?
5. Are you familiar with the *SEL Package-Based System Development Process* document?
6. For an upcoming COTS-based project, would you use the *SEL Package-Based System Development Process* document? If yes, why? If no, why not?
7. What parts of the process and/or the document would you improve and how?

Interview Guide 3: Additional COTS follow-up interviews

Who: COTS-based project leads

Subjects covered: Follow-up COTS information

Duration: 15 minutes

Note: This interview should also include reintroducing yourself and our study to the project leads. Mention that this final interview is to verify the data already collected and clarify any areas on which we need more information. For this interview, meet with the project lead and any other team members that you think appropriate to include to verify all the data collected on that project.

Interviewee(s):

Project(s):

Interviewer:

Scribe:

Date of interview:

Duration:

Location:

Date of initial interview:

Date of follow-up interview:

Before the interview,

- List the CTIFs that are on the Kano drive for that project
- Verify the matrix and supply any reasons why process steps were or were not followed

Bring to the interview,

- Matrix for that project
- Process characterization

Questions:

1. Have you completed CTIFs for each COTS or tool that you are currently using? [Definitely for all SEL projects, ask non-SEL project to also comply.] If not, fill in hardcopies of CTIFs during the interview with the project lead.
2. This is the process characterization that we have developed after interviewing projects. How representative is it of your project? [Take notes on areas that they believe they differ from the process characterization.]
3. These are the specific process steps that we noted during interviews. [Show matrix of steps versus interviews for that project.] Allow me to review the data that we have from you as to whether or not you followed a certain process step. Fill in YES for a project completed this step; fill in NO for a project not done this step. Give a simple reason for why the project completed or did not complete a step.

Appendix B. SEL Forms

This appendix includes the following SEL forms:

- WEF (October 1995)
- COTS WEF (June 1997)
- Newest WEF (November 1997)
- CTIF

B.1 WEF (October 1995)

WEEKLY EFFORT FORM

Use this form to record all hours you worked during the week.

Name:

Project:

Date (Friday):

For Librarian's Use Only	
Number:	
Date:	
Entered by:	
Checked by:	

Application or Subsystem (or "N/A" as appropriate)				
Release/Build Number (or "N/A" as appropriate)				
SCR Number (or "N/A" as appropriate)				
Hours By Activity (List hours in a separate column for each application, release/build, and SCR combination.)				
PREDESIGN	Requirements Spec. Definition/ Development	Hours spent defining and developing the requirements specifications		
	Requirements Analysis	Hours spent understanding requirements specifications or understanding SCRs for enhancements or adaptations		
	Error Analysis/ Debugging	Hours spent finding a known error in the system; may be in response to SFR, STR, SCR (includes generation and execution of tests associated with finding the error)		
	Impact Analysis/ Cost Benefit Analysis	Hours spent analyzing several alternative implementations and/or comparing their impact on schedule, cost, and ease of operation		
DESIGN	Design Creation or Modification	Hours spent developing or changing the system, subsystem, or component design (includes development of PDL, design diagrams, meeting materials, etc.)		
	Design Review/ Inspection	Hours spent reading or reviewing design (includes design meetings and consultations, as well as formal and informal reviews, walkthroughs, and inspections)		
CODE	Code Generation/ Modification	Hours spent actually coding system components (includes both desk and terminal code development)		
	Code Review/ Inspection	Hours spent reading code (for any purpose other than isolation of errors) or inspecting other people's code		
	Unit Testing	Hours spent testing individual components of the system (includes writing test drivers and informal test plans)		
TEST	System Integration/ Integration Testing	Hours spent integrating components into the system; hours spent writing and executing tests that integrate system components (includes system tests)		
	Regression Testing	Hours spent regression testing the modified system		
	Independent Testing Support	Hours spent supporting independent testing, including training of testers		
MISC	Prototyping	Hours spent prototyping to investigate a particular issue (not to be confused with other activity hours when the entire system is a prototype)		
	COTS/GOTS	Hours spent evaluating, selecting, procuring, integrating and testing COTS and GOTS products		
OTHER	Documentation	Hours spent creating and reviewing deliverable documents		
	Training for Self	Hours spent taking courses (including computer-based training), attending seminars, etc.		
	User Support/ Training	Hours spent training users and responding to their questions		
	Management	Hours spent managing or coordinating work and reporting status		
	Other	Other development hours not covered above		
	Total	Total hours per column	0.0	0.0
	Grand Total	Total hours	0.0	

November 1995

B.2 COTS WEF (June 1997)

Experimental "COTS modified" **WEEKLY EFFORT FORM**

For Librarian's Use Only
Number:
Date:
Entered by:
Checked by:

Use this form to record all hours you worked during the week.

Name:

Project:

Date (Friday):

Application or Subsystem (or "N/A" as appropriate)				
Release Number (or "N/A" as appropriate)				
Build Number (or "N/A" as appropriate)				
SCR Number (or "N/A" as appropriate)				
Hours By Activity (List hours in a separate column for each application, release/build, and SCR combination.)				
PREDESIGN	Requirement Spec. Definition/ Development	Hours spent defining and developing the requirements specifications		
	Requirements Analysis	Hours spent understanding requirements specs or understanding SCRs for enhancements or adaptations		
	Error Analysis/ Debugging	Hours spent finding a known error in the system; may be in response to SFR, STR, SCR		
	Impact Analysis/ Cost Benefit Analysis	Hours spent analyzing several alternative implementations and/or comparing their impact on schedule, cost, and ease of operation		
DESIGN	COTS/GOTS Evaluation	Hours spent in COTS/GOTS evaluation activities (i.e., identifying packages, collecting information, attending demos, evaluating and selecting COTS/GOTS packages)		
	Design Creation or Modification	Hours spent developing or changing the system, subsystem, or component design (includes development of PDL, design diagrams, meeting materials, etc.)		
	Design Review/ Inspection	Hours spent reading or reviewing design (includes design meetings and consultations, formal and informal reviews, walkthroughs, and inspections)		
CODE	COTS/GOTS Integration	Hours spent integrating COTS/GOTS (and other software components) to produce individual applications/subsystems (i.e., writing and debugging glueware, COTS package familiarization)		
	Code Generation/ Modification	Hours spent actually coding system components (includes both desk and terminal code development)		
	Code Review/ Inspection	Hours spent reading code (for any purpose other than isolation of errors) or inspecting other people's code		
	Unit Testing	Hours spent testing individual components of the system (includes writing test drivers and informal test plans)		
TEST	System Integration/ Integration Testing	Hours spent integrating components into the system; hours spent writing and executing tests that integrate system components (includes system tests)		
	Regression Testing	Hours spent regression testing the modified system		
	Independent Testing Support	Hours spent supporting independent testing, including training of testers		
MISC	Procurement	Hours spent procuring/purchasing, interacting with vendor regarding licensing/maintenance agreements, etc.		
	Prototyping	Hours spent prototyping to investigate a particular issue		
OTHERR	Documentation	Hours spent creating and reviewing deliverable documents		
	Training for Self	Hours spent taking courses (including computer-based training), attending seminars, etc.		
	User Support/ Training	Hours spent training users and responding to their questions		
	Configuration Management	Hours spent in configuration management		
	Management	Hours spent managing or coordinating work and reporting status		
	COTS/GOTS Other	Other COTS/GOTS specific hours not covered above		
	Other	Other development hours not covered above		
	Total	Total hours per column	0.0	0.0
	Grand Total	Total hours	0.0	

A. Parra, June 1997

B.3 Newest WEF (November 1997)

WEEKLY EFFORT FORM

Use this form to record all hours you worked during the week.

Name:

Project:

Date (Friday):

For Librarian's Use Only
Number:
Date:
Entered by:
Checked by:

Application or Subsystem (or "N/A" as appropriate)				
Release Number (or "N/A" as appropriate)				
Build Number (or "N/A" as appropriate)				
SCR Number (or "N/A" as appropriate)				
Hours By Activity (List hours in a separate column for each application, release/build, and SCR combination.)				
PREDESIGN	Requirement Spec. Definition/ Development	Hours spent defining and developing the requirements specifications		
	Requirements Analysis	Hours spent understanding requirements specifications or understanding SCRs for enhancements or adaptations		
	Error Analysis/ Debugging	Hours spent finding a known error in the system; may be in response to SFR, STR, SCR (includes generation and execution of tests associated with finding the error)		
	Impact Analysis/ Cost Benefit Analysis	Hours spent analyzing several alternative implementations and/or comparing their impact on schedule, cost, and ease of operation		
DESIGN	COTS/GOTS Evaluation	Hours spent in COTS/GOTS evaluation activities (i.e., identifying packages, collecting information, attending demos, evaluating and selecting COTS/GOTS packages)		
	Design Creation or Modification	Hours spent developing or changing the system, subsystem, or component design (includes PDL, design diagrams, meeting materials, etc.)		
	Design Review/ Inspection	Hours spent reading or reviewing design (includes design meetings and consultations, formal and informal reviews, walkthroughs, and inspections)		
CODE	COTS/GOTS Integration	Hours spent integrating COTS/GOTS (may be with other software components) to produce individual applications/subsystems (i.e., writing and debugging glueware)		
	Code Generation/ Modification	Hours spent actually coding system components (includes both desk and terminal code development)		
	Code Review/ Inspection	Hours spent reading code (for any purpose other than isolation of errors) or inspecting other people's code		
	Unit Testing	Hours spent testing individual components of the system (includes writing test drivers and informal test plans)		
TEST	System Integration/ Integration Testing	Hours spent integrating components into the system or writing and executing tests that integrate system components (includes system tests)		
	Regression Testing	Hours spent regression testing the modified system		
	Independent Testing Support	Hours spent supporting independent testing, including training of testers		
OTHER	COTS Package Familiarization	Hours spent learning to use a COTS package (not formal training, which would be listed under training for self; also does not include evaluation)		
	Prototyping	Hours spent prototyping to investigate a particular issue (not to be confused with other activity hours when the entire system is a prototype)		
	Training for Self	Hours spent taking courses (including computer-based training), attending seminars, etc.		
	User Support/ Training	Hours spent training users and responding to their questions		
	CM	Hours spent in configuration management		
	Procurement	Hours spent procuring/purchasing, interacting with vendor regarding licens-ing/maintenance agreements, etc.		
	Documentation	Hours spent creating and reviewing deliverable documents		
	Management	Hours spent managing or coordinating work and reporting status		
	COTS/GOTS Other	Other COTS/GOTS specific hours not covered above		
	Other	Other hours not covered above (i.e., department and all-hands meetings)		
	Total	Total hours per column	0.0	0.0
	Grand Total	Total hours	0.0	

November 1997

B.4 CTIF

COTS AND TOOLS INFORMATION FORM

Use this form to obtain context and evaluation data, verify at project completion. For each COTS product or tool, use a separate CTIF.

Name: **Date:**

Project:

COTS Product or Tool: **Version Number:** **Vendor:**

1. Reasons for using tool or COTS: Check all that apply.
   - ☐ requirements definition ☐ requirements analysis ☐ requirements ☐ tracking/traceability
   - ☐ design ☐ simulation/modeling ☐ code generation ☐ static analysis
   - ☐ compilation ☐ debugging ☐ configuration management
   - ☐ integration ☐ QA ☐ reengineering ☐ testing
   - ☐ reverse engineering ☐ change management ☐ project tracking ☐ documentation
   - ☐ training ☐ information management ☐ reuse management
   - ☐ measurement ☐ risk analysis ☐ communication ☐ project planning/estimation
   - ☐ application functionality

2. Support provided for tool or COTS: Check all that apply.
   - ☐ demos ☐ informal or partial documentation ☐ full documentation ☐ courses ☐ help desk

3. Activities supported by tool or COTS: Check all that apply.
   - ☐ requirements definition ☐ requirements analysis ☐ design ☐ coding ☐ testing
   - ☐ documentation ☐ CM ☐ QA ☐ management ☐ other

4. Usage frequency of tool or COTS: (Select one from choices below, enter letter of selected item here.)
   a. no usage b. used once or twice c. monthly d. weekly e. daily

5. Functionality of tool or COTS: (Select one from choices below, enter letter of selected item here.)
   a. no data available b. abandoned, due to lack of functionality
   c. major expected functions missing d. some expected functions missing
   e. most expected functions present f. all expected functions present

6. Usefulness of tool or COTS: (Select one from choices below, enter letter of selected item here.)
   a. no data available b. abandoned, due to problems c. many problems encountered
   d. some problems encountered e. few problems encountered f. no problems encountered

7. Impact of tool or COTS on project's success: (Select one from choices below, enter letter of selected item here.)
   a. impossible to estimate b. major negative impact c. some negative impact overall
   d. positive and negative impacts balance out e. some positive impact f. major positive impact

8. Is COTS or tool embedded in software, i.e., is COTS being delivered as part of the system? YES ☐ NO ☐

A. Parra, September 1997

Appendix C. Sample SEL Study Brief

Study Brief Number: 7

ISSUE: COTS Evaluation Team

PURPOSE: Document the SEL's understanding of the COTS Evaluation Team for the purpose of disseminating information to the FDF community and clarification for the SEL in regards to the COTS study.

CURRENT UNDERSTANDING:

The team was formed in 1995 to address a move toward COTS solutions in FDD. Originally part of Code 551, Flight Mechanics. Currently part of the Code 550 Flight Dynamics Technical Support Office (TSO).

Who is the Evaluation Team?

- Composed of problem domain experts and mission team members
- Led by Sue Hoge, GSFC analyst
- Matrixed on a as-needed basis, not dedicated full-time to evaluations

What are they doing?

- Evaluating COTS for Flight Dynamics mission planning and orbit determination
- Providing evaluation services to mission teams, as requested
- Providing independent software evaluations
- Monitoring new COTS products, as available, and maintaining data on products that meet specific domain needs
- Publishing evaluation reports
- Updating *Guidelines for Evaluating COTS at the FDF*, as needed

What process is followed?

- Basic process is outlined in *Guidelines for Evaluating COTS at the FDF*
- Establish the objectives of an evaluation
- Establish the evaluation type
- Determine the evaluation method
  - Basic/standard evaluation methods
  - Variations on standard evaluation methods
- Establish evaluation criteria
- Perform evaluation
- Document results
- Perform benchmarks, regression testing, follow-up evaluations

What tools have they evaluated?

- STK (AGI)
- PODS (AGI)
- GEODYN(Code 900, GOTS)
- OASYS(ISI)
- PROBE(BBN)
- PATTERN (BBN)
- GREAS (AGI)

What problems have been encountered?

Public awareness of Evaluation Team and services is low.

What else was learned about the Evaluation Team?

- They are building an "experience base" of COTS evaluations (for multiple products, multiple missions).
- Guidelines document
  - Domain specific, and not intended to be a general methodology for any COTS software evaluation
  - Working document with lessons learned mixed in with process
  - Written from hands-on perspective

What do we suggest?

The COTS product evaluation questions from Table 3 of *SEL Packaged-Based System Development* (page 22) are valid in the COTS Evaluation Team environment. We recommend that the SEL distribute the modified COTS product evaluation questions (addition of two questions suggested by Sue Hoge) as a one-pager to technical personnel. We also recommend that the Evaluation Team use modified COTS product evaluation questions as part of its process because these are issues that Sue Hoge typically addresses with Evaluation Team.

FEEDBACK: None available at this time, E-mail comments to responsible author.

ORIGINAL AUTHOR(S): Amy Parra and Steve Kraft

RESPONSIBLE AUTHOR: Amy Parra

CONTRIBUTOR(S): Sue Hoge

REFERENCES/RELEVANT LINKS:

- Guidelines for Evaluating COTS at the FDF document
- STK Evaluation and Test Results
- STK PODS Evaluation Final Report
- OASYS Evaluation Report
- Interview notes (from two COTS study interviews with Sue Hoge)

HISTORY: Study brief published 11/11/97

References

1. Waligora, S., *SEL Packaged-Based System Development Process*, 1996
2. SEL-81-305, *SEL Recommended Approach to Software Development*, Revision 3, June 1992
3. Glaser, B. G. and A. L. Strauss, *The Discovery of Grounded Theory: Strategies for Qualitative Research*, Aldine Publishing Company, 1967
4. Miles, Matthew B. and A. Michael Huberman, *Qualitative Data Analysis: An Expanded Sourcebook*, 2nd Edition, Sage Publications, 1994
5. SEL-85-004, *Evaluation of Software Technologies: Testing, Cleanroom, and Metrics*, May 1985
6. Seaman, C. B. and V. R. Basili, "An Empirical Study of Communication in Code Inspections," *Proceedings of the 1997 International Conference on Software Engineering*, Boston, MA, May 17-24, 1997

Acronyms

COTS	commercial off-the-shelf
CSC	Computer Sciences Corporation
CTIF	COTS and Tools Information Form
FDD	Flight Dynamics Division
GOTS	Government off-the-shelf
GQM	goal/question/metric
GSFC	Goddard Space Flight Center
IMACCS	Integrated Monitoring, Analysis, and Control COTS System
NASA	National Aeronautics and Space Administration
ORR	operational readiness review
SDR	system design review
SEL	Software Engineering Laboratory
SRR	system requirements review
WEF	Weekly Effort Form